МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение
высшего образования
«Национальный исследовательский ядерный университет
«МИФИ»

Лабораторная работа №2:
«Поиск скрытого сообщения в потоке трафика»

По дисциплине «Способы построения скрытых каналов»

Выполнила студент группы Б19-515
Щербакова Александра

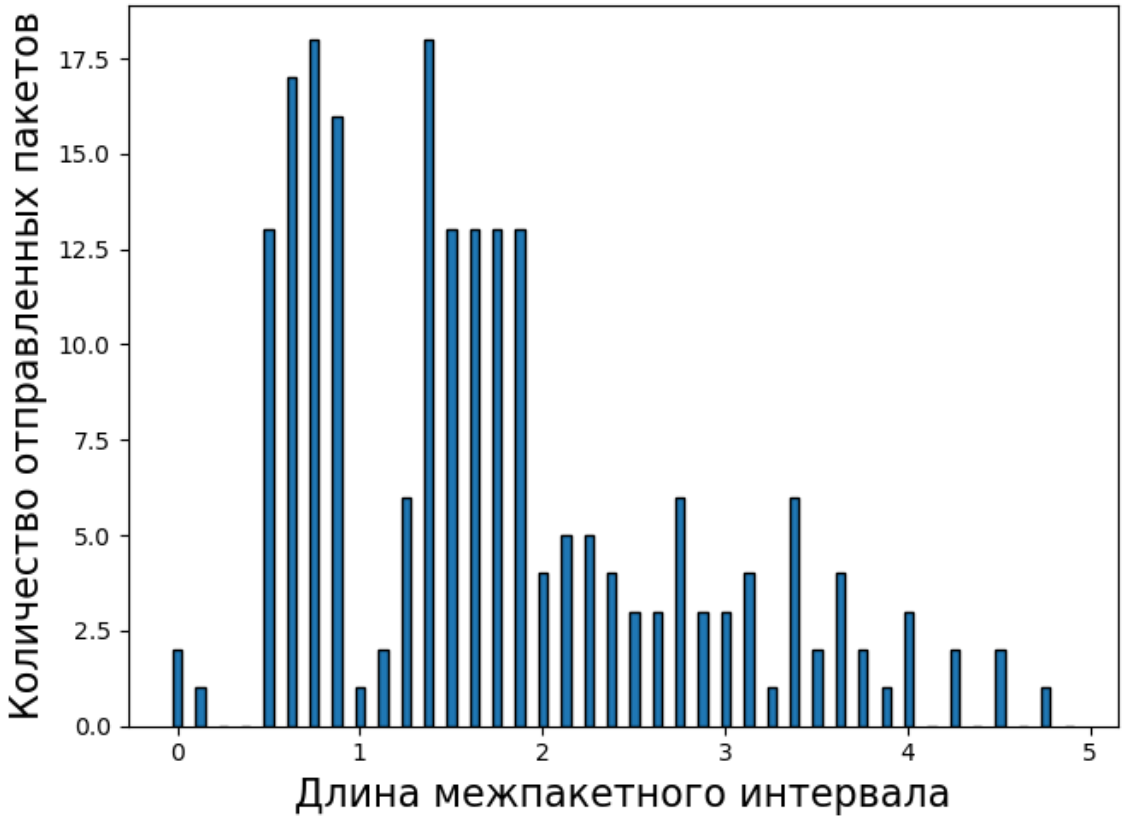
Москва, 2023 г.

№ по списку группы - 18; 18 mod 12 + 1 = 7 вариант задания

Задание 1

Дан дамп трафика. Известно, что при взаимодействии двух сторон было передано скрытое сообщение.

1) *Построить гистограмму отсортированных по возрастанию длин межпакетных интервалов - код в приложении А.*




```
aleksandrishche@vb:~/fin$ python3 lab2_1.py
Всего пакетов: 211
Максимальный временной интервал: 5.00756299999999
Средняя длина межпакетного интервала: 1.7938447535545023
Максимум пакетов со схожей длиной межпакетного интервала: 18
Пакетов со средней длиной межпакетного интервала: 13
Вероятность присутствия скрытого канала: 27%
aleksandrishche@vb:~/fin$
```

2) *Определить значение вероятности присутствия скрытого канала в системе согласно формуле:*

$$P = 1 - \lim_{N \to \infty} \frac{C_\mu(N)}{C_{max}(N)}. \quad (1)$$

где $C_\mu(N) = 13$ - количество пакетов со средней длиной межпакетных интервалов; $C_{max}(N) = 18$ - наибольшее число пакетов с определенной длиной межпакетных интервалов.

$P = (1 - \frac{13}{18}) \cdot 100\% = 27,8\%$ - *вероятность присутствия скрытого канала.*

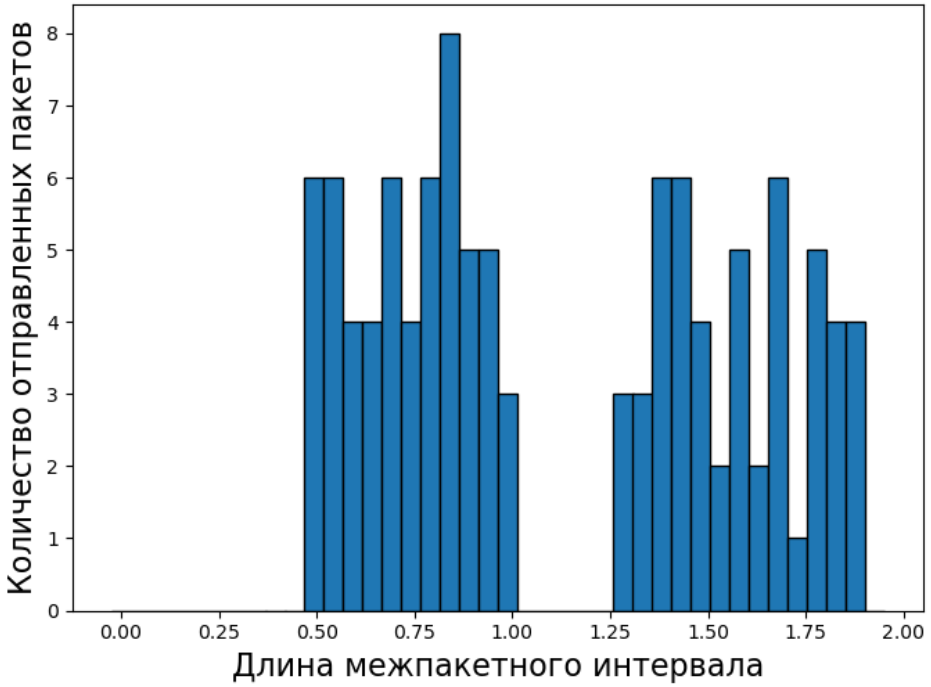
Задание 2

По новой информации, скрытое сообщение передается, начиная с ~100 пакета и до последнего пакета в дампе трафика.

1) *Построить гистограмму отсортированных по возрастанию длин межпакетных интервалов для трафика, в который включен только промежуток с передачей скрытого сообщения;*

По сравнению с первым кодом надо изменить только функцию *read_file* так, чтобы в переменную с данными были записаны только пакеты, начиная с сотого. Код новой версии функции *read_file* в приложении Б.

```
aleksandrishche@vb:~/fin$ python3 lab2_2.py
Всего пакетов: 112
Максимальный временной интервал: 1.9749010000000453
Средняя длина межпакетного интервала: 1.183711107142857
Максимум пакетов со схожей длиной межпакетного интервала: 8
Пакетов со средней длиной межпакетного интервала: 0
Вероятность присутствия скрытого канала: 100%
aleksandrishche@vb:~/fin$
```



2) *Определить значение вероятности присутствия скрытого канала в системе согласно формуле (1):*

Аналогично, как в первом задании:

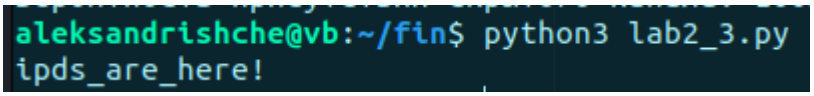
$P = (1 - \frac{0}{8}) \cdot 100\% = 100\%$ - *вероятность присутствия скрытого канала.*

Задание 3

Разработать средство декодирования скрытого сообщения и определить, какое скрытое сообщение было передано.

На основе данных, полученных на предыдущем шаге, было сделано предположение о том, что все пакеты с межпакетным интервалом из левого пика кодируют ноль, а пакеты из правого пика – единицу. После этого предположения была получена бинарная строка, которая потом была переведена в текст (каждые 8 бит кодируют один символ). Код в приложении В.

Результат: “*ipds_are_here!*” - скрытое сообщение.



Заключение

В рамках лабораторной работы изучался метод обнаружения скрытого канала по времени, работающего на основе изменения длин межпакетных интервалов.

1) *Сделать вывод о применимости метода обнаружения в случае малого объема переданной скрытно информации.*

Результаты работы демонстрируют, что метод действительно применим для малого объема данных, хоть и с условностями - необходима информация о том, начиная с какого пакета передается скрытое сообщение.

В ситуации, когда объем передаваемой скрытно информации слишком мал по сравнению со всем объемом трафика, обнаружить скрытый канал невозможно, поскольку пики пакетов, отвечающих за нули и единицы, могут затеряться среди всего объема трафика, и будет один пик, как если бы это было нормальное распределение.

2) *Предложить способ первоначальной обработки трафика в случае, если объем трафика большой, а количество переданной скрытно информации - малый.*

Для возможности обнаружить скрытый канал при большом объеме трафика и маленьком объеме скрытно передаваемой информации, можно использовать метод «скользящего окна». Суть этого метода заключается в том, чтобы сначала рассматривать первые, например 1000 пакетов, затем сделать шаг в 100 пакетов и рассматривать с 100 до 1100 и т.д.. Таким образом, правильно подобрав параметры скользящего окна, будет возможность обнаружить скрытый канал.

3) *Предложить способ расширения метода обнаружения на мультисимвольные скрытые каналы на основе изменения длин межпакетных интервалов.*

При мультисимвольном скрытом канале пиков на графике будет равно количеству различных передаваемых символов. При этом количество пакетов в пиках будет связано с вероятностью появления конкретного символа в тексте. Значит, в формуле вероятности наличия скрытого канала нужно будет учитывать различные Cμ, поскольку их необходимо будет считать для каждых двух соседних пиков.

Приложение А

```
import csv
import numpy as np
import matplotlib.pyplot as plt

def read_file(file_name):
    result = []
    with open(file_name, encoding='utf-8') as r_file:
        dump = csv.reader(r_file, delimiter=",")
        count = 0
        for row in dump:
            if count != 0:
                result.append(row[1])
            count += 1
    return result


if __name__ == "__main__":

    # достаем метки времени из дампа
    timestamps = read_file('dump_7.csv')

    # высчитываем интервалы
    intervals = [float(timestamps[i + 1]) - float(timestamps[i]) for i
in range(len(timestamps) - 1)]

    # смотрим распределение длин межпакетных интервалов
    distributed_intervals = []
    counts = []
    intervals_amounts = 40
    step = max(intervals) / intervals_amounts
    # будем также смотреть среднюю длину межпакетного интервала
```

```
    average_interval = np.mean(intervals)
    index_of_average = 0

    print("Всего пакетов: " + str(len(intervals)))
    print("Максимальный временной интервал: " + str(max(intervals)))
    print("Средняя длина межпакетного интервала: " +
str(average_interval))

    count = 0
    for i in range(intervals_amounts):
        # сами интервалы
        distributed_intervals.append(count)
        # инициализируем нулями массив распределений
        counts.append(0)
        # запоминаем индекс интервала, наиболее похожего на средний
        if count + step > average_interval and count <=
average_interval:
            index_of_average = i
        count += step


    # заполняем массив распределения
    for i in range(len(intervals)):
        for j in range(len(distributed_intervals) - 1):
            # добавляем счетчик интервалу, который попал в текущий
            if distributed_intervals[j] < intervals[i] and intervals[i]
< distributed_intervals[j + 1]:
                counts[j] += 1
                break

    print("Максимум пакетов со схожей длиной межпакетного интервала: " +
str(max(counts)))
    print("Пакетов со средней длиной межпакетного интервала: " +
str(counts[index_of_average]))
    print("Вероятность присутствия скрытого канала: " + str(int((1 -
counts[index_of_average]/max(counts)) * 100)) + "%")

    fig, ax = plt.subplots()
    plt.xlabel("Длина межпакетного интервала", fontsize=16)
    plt.ylabel("Количество отправленных пакетов", fontsize=16)
    ax.bar(distributed_intervals, counts, width=0.05, edgecolor='black')
    plt.show()
```

Приложение Б

```
def read_file(file_name):
    result = []
    with open(file_name, encoding='utf-8') as r_file:
        dump = csv.reader(r_file, delimiter=",")
        count = 0
        for row in dump:
            if count > 99:
                result.append(row[1])
            count += 1
    return result
```

Приложение В

```
import csv
import matplotlib.pyplot as plt


def read_file(file_name):
    result = []
    with open(file_name, encoding='utf-8') as r_file:
        dump = csv.reader(r_file, delimiter=",")
        count = 0
        for row in dump:
            if count > 99:
                result.append(row[1])
            count += 1
    return result


if __name__ == "__main__":
    # достаем метки времени из дампа
    timestamps = read_file('dump_7.csv')

    # высчитываем интервалы
    intervals = [float(timestamps[i + 1]) - float(timestamps[i]) for i
in range(len(timestamps) - 1)]

    covert_message = ''
    for i in range(len(intervals)):
        if intervals[i] < 1.1:
            covert_message += '0'
        else:
            covert_message += '1'
```

```
string = ''
for i in range(0, len(covert_message), 8):
    string += chr(int(covert_message[i:i + 8], 2))

print(string)
```